REMARKS

Claims 1-3, 5 and 7-22 are pending. Claims 1, 10, 16 and 19 are independent.

The examiner maintained his rejections of independent claims 1, 10, 16 and 19 under 35 U.S.C. §101.

With respect to applicant's independent claims 1 and 19, which recite "[a] computer program product residing on a computer readable device ...," applicant contends that such claims are directed to subject matter held to be patentable by the Federal Circuit. As stated in MPEP 2106.01:

> In contrast, a claimed computer-readable medium encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. See *Lowry*, 32 F.3d at 1583-84, 32 USPQ2d at 1035. Accordingly, it is important to distinguish claims that define descriptive material *per se* from claims that define statutory inventions.

Further, the Patent Office consistently and repeatedly issues patents directed to "a computer program product residing on a computer readable ...," including, for example, recently issued U.S. patent Nos. 7,216,336 and 7,212,882. Applicant thus contends that independent claims 1 and 19 recite patentable subject matter.

Independent claims 1, 10, 16 and 19, recite a branch instruction that causes a data processing apparatus and/or processor to evaluate a branch condition relating to whether a state, of a state name indicating the availability of a resource, is a specified value, and to branch to a specified address based on that evaluation. Applicant's contends that the independent claims are directed to a branching operation that is tangible, in that the claims are all tied to a particular machine (namely, a data processing apparatus or processor), concrete, in that the recited branching operation is repeatable and produces the same results upon different performances of the recited operation (e.g., branching will occur each time that the state of the state name specified in the branch instruction is a specified value), and useful, in that a beneficial result is produced for operation of such machine.

The examiner also stated in the May 4, 2007, Final Action:

> 12. Furthermore, it produces no tangible result because the computer program product residing on a computer readable storage device to

> cause to branch if a state is at a specified value is an intended use, therefore an abstract idea. The evidence shows applicant is claiming "control logic" (see claim 1, line 8). Logic is an abstract idea. (Final Action, pages 3-4)

In response, applicant amended independent claims 1, 10, and 19 to call for control logic "circuitry."

Applicant respectfully requests reconsideration and withdrawal of the examiner's rejections of independent claims 1, 10, 16 and 19 under 35 U.S.C. §101.

The examiner continues to reject claim 1 on the ground of non-statutory obviousness-type double patenting as being unpatentable over claim 29 of U.S. Patent No. 6,668,317.

Applicant defers responding to the examiner's double-patenting rejection until the allowability of the subject matter claimed herein is indicated.

The examiner continues to reject claims 1-20 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,724,563 to Hasegawa in view of U.S. Patent No. 6,223,208 to Kiefer.

In addition, the examiner continues to reject claims 1 and 19 under 35 U.S.C. §103(a) as being unpatentable by U.S. Patent No. 4,454,595 to Cage in view of Kiefer. Further, the examiner rejected claims 1, 10 and 19 under 35 U.S.C. §103(a) as being unpatentable by U.S. Patent No. 6,275,508 to Aggarwal in view of Kiefer.

Applicant amended independent claims 1, 10, 16 and 19 to recite a feature, similar to the feature previously recited in claim 21, that the state indicates the availability of a queue resource of the data processing apparatus. Applicant cancelled claim 21, and amended claim 22 to correct claim 22's dependency.

In rejecting claim 1 as being obvious over Hasegawa in view of Kiefer, the examiner stated in the non-final action of December 22, 2006:

> 20. As to claims 1, Hasegawa disclosed a decisions on the direction of the instruction processing based on the state or condition (see also fig-s, and fig.10, see also the implicit value Z or C encoded in opcode in Table 1, col.1, lines 41-52, see the specific structure of also the teaching of Z and C flags set and reset in col.11, lines 14-40, lines 54-67, col.12, lines 1-5).

> 21. Hasegawa taught a data processing system including a branch instruction that caused an execution of instruction stream to branch to an instruction at an address (x) specified in the instruction if a state, of a specified name (Z,C), indicating the availability of a resource (see the branch instruction format in fig.2, see the value specified in the branch instruction field , see also fig.5, and fig.10, see also the implicit value Z or C encoded in opcode in Table 1 , col.11, lines 41-52, see also the teaching of Z and C flags set and reset in col.11 , lines 14-40, lines 54-67, col.12, lines 1-5). the parallel processor, see the pipeline processor in fig.6).
>
> 22. Hasegawa did not specifically show the process of a plurality of threads as claimed. However, Kiefer taught the microengine (see fig.2 [200]) having a control logic [context switching] and execution logic including the arithmetic unit [260]-[250] and general purpose unit [register set], and configured to process a plurality of threads (see the general purpose register and the execution of threads in col.8, lines 4-45). It would have been obvious to one of ordinary skill in the art to use Kiefer in Hasegawa for including the plurality of threads as claimed because the use of Kiefer could increase the processing bandwidth of Hasegawa, and it could be done by predefining the plurality of threads into Hasegawa with modified configuration variables (e.g. thread length and thread type), and because Hasegawa taught a pipeline processing for processing a plurality of instructions in parallel by dividing the execution process into a plurality of processing stages (see col.1, lines 10-17), which was recognizable by one of ordinary skill in the art to use threads, or processing stages, as claimed in order to increase the processing bandwidth (e.g. the parallel processing), and in doing so, provided a motivation. (December 22, 2006, Office Action, pages 9-10, paragraphs 20-22)

Applicant respectfully disagrees with the examiner's contentions.

Applicant's independent claim 1 recites "branch to another instruction of the instruction stream at an address specified in the branch instruction if a state, of a state name specified in the branch instruction is a specified value, with the state indicating the availability of a queue resource of the data processing apparatus, wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads." Thus, the state indicating the availability of a queue resource is made available to all the microengines so that the microengines can determine if the resource is being used by another microengine.

Hasegawa discloses a pipeline processor that can execute predictive branch instructions (Abstract). While Hasegawa's processor includes a register file 12 and an execution section 11, at no point does Hasegawa disclose that a plurality of such processors are used, each of which being configured to process a plurality of threads. Because Hasegawa's architecture does not include a plurality of microengines each configured to process a plurality of threads, Hasegawa's processor cannot make a state, indicating the availability of a queue resource, available to a plurality of microengines. Indeed, there would be no reason for Hasegawa's processor to monitor the state indicating the availability of a resource, such as a queue resource (applicant notes that Hasegawa does not even mention "queue") because Hasegawa does not have multiple processors competing for the same resource. Accordingly, Hasewaga does not disclose or suggest at least "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

Kiefer describes a multithreaded processor that uses idle register/storage functional units in a processor core to dynamically exchange threads (Kiefer, col. 1, lines 7-9). Specifically, Kiefre describes its processor core 200 as follows:

> **Returning now to FIG. 2, processor core 200 comprises at least an instruction register/control 230, a context switch unit 240, several register/register (RR) units 250, several register/storage (RS) units 260, and thread register sets 270, 272, 274. Inside each thread register set 270, 272, 274 are a number of specialized registers and components; the minimum being a general purpose register, one or more special purpose registers, a multiplier, a floating point register, and an arithmetic logic unit. Register/register units 250 and register/storage units 260 generally access the general purpose registers and the floating point registers from the executing thread register set 270. Register/register execution units 250 are operatively connected to receive instructions from the instruction register/control 230 and to transfer executed instruction results which do not require access to the memory system 220 into and out of the executing thread register set 270. Two register/register units 250 are illustrated in the embodiment shown in FIGS. 2 and 3, however, there can be as few as one or as many of these units as necessary or as defined by the computer architecture. Similarly, there can be few or numerous register/storage units 260 which perform instructions including those instructions that access the memory system 220. For the lazy context switching of the invention the**

> **register/storage units 260 read and write data from the thread register buffer sets 270, 272, and 274 and memory system 220, as enabled by the instruction register/control 230 and the context switch unit 240. Register/storage units 260 also receive normal instructions from the instruction register/control 230. Memory system 220 comprises the locations where the thread register states reside when not in the processor core 200. The state of a thread refers to the contents of the register files necessary for an instruction set capable of independent execution, called a thread, to execute. Of course, the state of a thread depends upon the processor architecture and implementation and, as an example only, a state can include the contents of a general purpose register file, a floating point register file, and other registers critical to the correct operation of the processor such as condition code registers, machine state registers, link and count registers, next instruction address register, exception registers, etc. (col. 8, lines 5-45)**

Therefore, Kiefer does not describe a configuration that includes a plurality of processors such as processor 200. Further, because Kiefer does not describe a plurality of processors, it also does not describe that a state of a specified state name is made available to such a plurality of processors. Indeed, it would pointless to monitor a state, indicating the availability of a queue resource, because Kiefer does not have multiple processors competing for that resource. (For that matter, Kiefer does not even mention a "queue".) Accordingly, Kiefer also fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

Because neither Hasegawa nor Kiefer discloses or suggests, alone or in combination, at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," applicant's independent claim 1, and the claims that depend from it, are therefore patentable over the prior art.

Applicant further notes that the examiner previously relied on U.S. patent No. 5,056,015 to Baldwin as allegedly teaching a queue. However, Baldwin describes "[a] multiprocessor subsystem, wherein each processor is separately microcoded so that the processors can run

concurrently and asynchronously" (see Abstract). Thus, Baldwin's processors do not appear to coordinate operations with each other, and thus Baldwin too does not need to make a state indicating the availability of a resource (such as a queue resource) available to Baldwin's processors. Baldwin too, therefore, fails to disclose or suggest "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads,"

As noted, the examiner also separately rejected independent claim 1 as being unpatentable over Cage in view of Kiefer, and also as being unpatentable over Aggarwal in view of Kiefer.

Cage describes a buffer for use with a word processing disk controller. Nowhere does Cage describe that any of the processors are multithreaded processors. Cage also does not describe that a state of a particular state name is made available to multithreaded processors. Cage also fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

Aggarwal describes a system for processing datagram headers during traverses from one interface of a networking device to another (Abstract). Aggarwal's system includes a sequencer unit that controls the selection of data from input data stream (FIGS. 1 and 12, and Abstract). The sequencer unit is controlled by a word instruction called the Write Control Store (WCS) (see col. 3, lines 61-63).

Aggarwal, however, does not describe any processors, let alone multithreaded processors, and certainly does not describe a plurality of multithreaded processors. Aggarwal, therefore, also does not describe a state of a particular state name that is made available to a plurality of multithreaded processors. Accordingly, Aggarwal fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality

microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

As explained above, Kiefer also does not disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads."

Because none of Aggarwal, Cage and Kiefer discloses or suggests, alone or in combination, at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," applicant's independent claim 1, and the claims that depend from it, are therefore patentable over these prior art references.

Claims 2-3, 5, 7-9 and 21-22 depend from independent claim 1 and are therefore patentable for at least the same reasons as independent claim 1.

Independent claims 10, 16 and 19 recite "evaluating a value of a state name specified in a branch instruction, the value of the state name indicating the availability of a queue resource of the processor; and performing a branching operation to cause an instruction stream to branch to another instruction of the instruction stream at an address specified in the branch instruction based on the value of the specified state name being set or cleared, wherein the state is available to the multiple microengines, each of the multiple microengines having control logic circuitry and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," or similar language. At least these features are not disclosed by the prior art cited by the examiner for reasons similar to those provided with respect to independent claim 1.

Accordingly, independent claims 10, 16, and 19, and the claims respectively depending from them, are patentable over the prior art.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

In view of the foregoing, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

No fee is believed due. Please apply charges to deposit account 06-1050, referencing attorney docket 10559-306US1.

Respectfully submitted,

Date: July 12, 2002

Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

PTO Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906